METHOD AND SYSTEM FOR REDUCING FALSE ALARMS IN NETWORK FAULT MANAGEMENT SYSTEMS

COPYRIGHT NOTICE

BACKGROUND OF THE INVENTION

The invention disclosed herein relates to computer network fault management. In particular, the present invention relates to improved techniques for reducing false alarms in such systems by a finer correlation of variables.

The expense of service downtime, the limited supply of network engineers, and the competitive nature of today's marketplace have forced service providers to rely more and more heavily on software tools to keep their networks operating at peak efficiency and to deliver contracted service levels to an expanding customer base. Accordingly, it has become vital that these software tools be able to manage, monitor, and troubleshoot a network as efficiently as possible. An important aspect of such troubleshooting is the detection and analysis of network faults and their causes.

A variety of software is currently available that improves network management through automated fault analysis. For instance, the Netcool®/Visionary™ software available from Micromuse Inc. evaluates a network's health by correlating data gleaned from various network devices in accordance with a set of expert system rules. Each rule defines which data

Express Mail No. EL718275755US

items or indicators, when detected together, indicate the presence or likelihood of a fault. For example, in formulating a diagnosis that a router's CPU is over-utilized, the software correlates relevant data that may have caused the problem, such as instability from a particular routing peer, poor access list configuration, and a forgotten debug setting.

System or device data can be correlated in a number of ways. For example, in the Netcool®/Visionary™ software, a window of time is divided into time slices and rule-based correlation is performed for each time slice. In each time slice, the software determines which of the indicators being monitored has reached a state of severity, and computes a percentage reflecting the number of such severe indicator states over the total number of indicators being monitored by the rule. If the result of the correlation is greater than a predetermined threshold percentage, the software marks the time slice as a positive result for the fault, an activity sometimes referred to herein as firing. If the rule fires for more than a threshold percentage of time slices during the time window, an alert or alarm is sent to a network management platform that the fault is likely to be occurring. For example, a rule with threshold percentage of 50% and a window size of 60 seconds, divided into twelve 5-second time slices, formulates a problem diagnosis when the rule has fired 6 or more times during the window.

The ability of software such as the Netcool®/Visionary™ program to predict problems and prevent them before they affect service uptime largely rests on the accuracy of the multivariable correlation. Accordingly, much effort goes into formulation of the rules and the selection of an appropriate set of indicators as relating to each type of fault being analyzed. However, even the best rules-based detection systems suffer from inherent problems arising from the generalized association of events and faults and the complex nature of large networks or other systems in which many related and unrelated events are occurring with great frequency.

Express Mail No. EL718275755US

Thus, rules-based correlation has a tendency to result in false alarms. For example, using the correlation techniques discussed above, while the intermittent positive detection of a severe condition in the three indicators considered by a rule may result in the rule firing and sending an alarm, the conditions may in fact be unrelated and coincidental and may in fact not be causing or otherwise related to the fault about which the alarm is sent. False alarms require the attention of service provider administrators and divert the resources needed to attend to real faults.

There is therefore a need for improved techniques for limiting the number of false alarms occurring during fault detection correlation.

SUMMARY OF THE INVENTION

It is an object of the present invention to solve the problems described above with existing fault correlation systems.

It is another object of the present invention to reduce the number of false alarms generated by fault detection systems.

It is another object of the present invention to improve the accuracy of and reduce noise in fault detection systems.

The above and other objects are achieved by improved techniques for correlating a number of variables or indicators in a rules-based correlation system. The techniques involve correlating over time the edges or transitions of indicators so that indicators which are not changing together are less likely to be correlated by a rule. Thus, software utilizing the improved techniques detects not only whether multiple variables are at given severities during each timeslice of a window, but also whether and how the variables changed severity from the prior timeslice. For example, if two or more indicators being monitored by a rule change state during

a time slice, but one transitions from a positive state to a negative state and the other indicator transitions from negative to positive state, then an edge or transition factor is generated which makes the rule less likely to correlate the two indicators.

These improved techniques raise accuracy through noise reduction, but do not necessarily increase the fault detection rate. In other words, the presence of correlated edges does not make a rule any stronger than it would be if edge correlation was not considered, but the absence of edges having correlation should make the rule less likely to fire. Similarly, if the indicators of a rule would not cause the rule to fire, edge correlation should not cause it to fire. In a situation where the indicators changed to a positive severity state together, but there are no edges in the current window, the rule should not be less likely to fire. The complete absence of edges in a window should not count against the rule.

In some embodiments, edge correlation is an indicator specific attribute and is specified on a per indicator basis. For example, some rules use the presence of an indicator as part of the rule, and this indicator does not transition. As such, the indicator would detract from the rule's firing if edge correlation were applied to it, since other indicators are capable of transitioning. In these embodiments, indicators are assigned an edge relevance to limit or prevent their use in edge correlation.

The above and other objects of the present invention are also achieved through a method for improving diagnosis of a complex problem, such as a problem detected in a computer network or other environment where symptoms are multivariate and rapidly changing, wherein a plurality of indicators expected to relate to the network problem are correlated over a window of time. The window of time consists of a plurality of sequential time slices in each of which a state of each indicator is determined such as by probing or polling. The method involves

determining which indicator or indicators changed state during a first time slice and which indicator or indicators did not change state during the first time slice. This comparison may be done for a number of time slices or preferably all the time slices in the window. In some embodiments, a transition is determined through frequent polling of the state of the indicators. In other embodiments, the transition is found by comparing the state of each of the indicators in the first time slice in the window, such as at the end of the time slice, with the state of the respective indicators in a second time slice immediately preceding the first time slice.

The method further involves computing a time slice transition or edge factor based upon the number of indicators whose state changed and a number of indicators whose state did not change during the first time slice. This transition factor is then used to adjust the correlation of the indicators over the window of time. In some embodiments in which indicators are correlated based on their states in each time slice, the time slice transition factor is used to adjust the correlation by applying it to the specific correlation for the first time slice. When the transitional determination is made for multiple or all time slices, multiple transition factors are generated, one for each time slice, and each is applied to the correlation for the respective time slice.

In some embodiments, the indicators have a plurality of possible states including a low state and a high state, and determining changes in the states of the indicators involves determining which indicator or indicators changed state from low to high and which indicator or indicators changed state from high to low. This determination may then form the basis for computing the time slice transition factor by, for example, identifying a maximum among the number of indicators whose state did not change, the number of indicators whose state changed from low to high, and the number of indicators whose state changed from high to low, and

dividing the maximum by the total number of indicators. This results in a ratio of correlated transitions over total indicators. The ratio has a value necessarily no greater than one, thus supporting the result explained above that the use of edge or transition correlation decreases not increases the likelihood that indicators will be correlated.

The use of arbitrary boundaries for time slices and potential delays in measuring states of indicators may result in two or more indicators which actually changed state very close in time to be considered to have changed state in different time slices. Thus, in some embodiments, this problem of jitter is solved by adjusting the first time slice to account for potential delays in determining states of the indicators. This adjustment may be accomplished by using different lengths of time slices which reflect the measurement delays. Alternatively, this adjustment may be accomplished by determining which indicator, if any, changed state last in time during the first time slice, determining whether any other indicators changed state during a predefined period preceding the time of last indicator state change, and considering such other indicators whose state changed during the predefined period as having occurred during the first time slice. The predefined period may be the same interval as the interval of the time slices.

Objects of the present invention are also achieved by a method for reducing false alarms in a system for analyzing faults in devices. The system correlates a plurality of indicators over a window of time and generates alarms based upon the correlation, with the window of time having sequential time slices in each of which a state of each indicator is probed. The method involves detecting whether and how transitions occurred in the states of the indicators during a first time slice. A time slice transition factor is computed that relates the number of the indicators whose state changed in similar fashion or did not change during the first time slice to the total number of indicators, and this transition factor is used to adjust the correlation of the

indicators over the window of time. If the factor reflects a lack of full correlation in the transitions during the first time slice, the correlation of the indicators is reduced, thus reducing the likelihood that a false alarm will be generated.

BRIEF DESCRIPTION OF THE DRAWINGS

The invention is illustrated in the figures of the accompanying drawings which are meant to be exemplary and not limiting, in which like references are intended to refer to like or corresponding parts, and in which:

Fig. 1 is a block diagram showing a network management system including an edge correlation component in accordance with embodiments of the present invention;

Fig. 2 is a timing diagram illustrating concepts underlying the present invention;

Fig. 3 is a flow chart showing a process of using edge correlation to reduce false alarms in accordance with embodiments of the present invention;

Figs. 4A-4B contain a flow chart showing the process of Fig. 3 is greater detail in accordance with embodiments of the present invention;

Fig. 5 is a flow chart showing a process of correlating edges of groups of indicators in accordance with embodiments of the present invention; and

Figs. 6-11 are exemplary data structures of indicator states over time and corresponding correlation factors which illustrate applications of embodiments of the present invention.

DETAILED DESCRIPTION OF THE PREFERRED EMBODIMENTS

Preferred embodiments of the present invention are now described with reference to the drawings in Figs. 1-11. Referring to Fig. 1, a system is shown for detecting the occurrence of faults in a network in accordance with the present invention. The system includes a network

management platform 10 which consolidates, manages and presents event information relating to a network 12 to network administrators. An exemplary network management platform 10 is the NETCOOL® suite of programs available from Micromuse Inc. The system further includes a set of status monitors 14 which poll devices 16 and software applications 18 connected or operating over the network 12. Devices 16 include many types of hardware devices such as CPUs, routers, printers, network interface cards, etc., and applications include many types of programs such as operating systems, server programs, browsers, security systems, etc.

The status monitors 14 collect many raw data values from industry standard SNMP management information databases within each managed or monitored network element on a substantially continuous basis, e.g., every second or every few seconds. The status monitors 14 normalize the raw data values as described below and feed the collected, normalized data to a set of event filters 20, sometimes referred to as or forming part of brokers, which manage the exchange of data collected by status monitors to other components of the system. Each status monitor uses a configuration file that specifies what SNMP objects it should monitor, how often it should poll those objects, and how it should interpret the results. The objects being monitored or instances of those objects are sometimes referred to herein as indicators, although indicators may include further information such as data on how to interpret the value of an object. The event filters 20 reduce the overall number of events detected by the status monitors 12 by isolating and sending only events that indicate a change in severity level in the monitored device or application.

The event data filtered and transmitted by the event filters 20 is received and processed by one or more correlation engines 22. The correlation engine 22 receives the normalized, filtered event data, either from the event filters 20 or directly from the status

monitors 14 through a subscription mechanism, and performs high-speed consolidation, correlation and analysis of the information to identify whether problems are occurring or likely to occur in the network. This analysis is performed by retrieving one or more rules from a rules database 24. The rules dictate which indicators or properties of the devices 16 or applications 18 being monitored by the status monitors 14, when found to be occurring together, are symptomatic of a given defined problem or fault. Rules are typically developed based on historical experience or other substantive expertise.

The rules are used by the correlation engine 22 to determine whether an alert or alarm should be sent to a network administrator informing him or her of the detection of the problem. When the correlation engine 22 receives the configuration files from event filters 20 and locates the rules which use the data, the engine 22 determines which indicators it needs data for in order to invoke the rules, and polls the status monitors 14 to subscribe to the event data for these indicators. The correlation engine 22 may also receive some event data as part of the configuration files. In some embodiments, the event filters 20 may also consult the rules to determine which event data received from status monitors 14 to filter and which to pass along to the correlation engine 22.

In accordance with the invention, the correlation engine 22 also includes an edge correlation routine 26 which improves the accuracy of the rule-based correlation performed by the correlation engine 22 and reduces the number of false alarms generated by the engine 22. The edge correlation routine 26, described in greater detail in the remainder of this specification, analyzes trends or transitions in the states of the indicators examined by the rule to determine how closely the various indicators are tracking to one another. Thus, while the correlation engine 22 may use any known technique to correlate event data relating to a number of devices

or applications, the edge correlation routine 26 computes an edge or transition factor which is applied to the correlation produced by the correlation engine.

A configuration tool 28 is provided in the system to allow administrators to manipulate the system and perform administrative functions such as adding or deleting filters, devices, or applications and starting and stopping the correlation engine. The configuration tool 28 also allows administrators to view results and modify settings controlling the operation of the system components, such as the severity levels used in filtering data or the threshold values used in generating alarms. This configuration tool 28 may be directly coupled to and in control of the correlation engine 22, or may be part of a system-wide configuration subsystem coupled to the network management platform 10 and used in controlling not only the correlation engine but other products forming part of the network management platform 10.

Each of the components shown in Fig. 1 may be implemented in computer hardware, software, or some combination of the two. For a component implemented as a software program, the program is stored in a memory device readable by a computer to cause the computer to perform the functions and methods described herein. Two or more of the components may be part of a single overall system, or may be remotely located and communicate with one another over any telecommunication network or system and using any desired protocol, including communication over the Internet, wireless technology, satellite, cable systems, etc.

The basic methodology of reducing false alarms through edge correlation is now described with reference to the illustration in Fig. 2 and the flow chart in Fig. 3. As shown in Fig. 2, an exemplary rule used to diagnose a given problem involves analysis of three indicators identified as indicators 1, 2 and 3. In this simplified illustration, each indicator may have two possible states - 0 and 1 - with one of the states representing a severity condition and the other

representing a normal operating condition. As shown in the illustration, indicator 1 remains at state 1 while indicators 2 and 3 change state over time. If the rule provides that the three indicators be collectively at a state of 1 for more than half the time, the correlation engine would determine from this illustration that the rule is satisfied and thus an alert should be generated.

However, a closer examination of Fig. 2 reveals that in fact the states of the indicators may not be sufficiently related to draw a conclusion that the problem defined by the rule is occurring. For example, indicator 1 is always at state 1 and does not transition as the other indicators do. Also, while indicators 2 and 3 transition together in times 1 and 2, they transition in different fashion in times 3, 4, and 5. This lack of coincidence in the transitions of the three indicators over time shows that, while as a general matter as defined by the rule the presence of these three indicators is indicative of a problem, in this case their concurrence may in fact be unrelated or caused by separate, unconnected conditions. Thus, the indicators should not necessarily be correlated and an alarm need not be sent.

Referring then to Fig. 3, a process in accordance with the invention involves monitoring devices to determine indicator states, step 40. The correlation engine computes a correlation factor based on the states of the indicators and the stored rule, step 42. The edge correlation routine compares the transitions of the indicator states over multiple discrete time periods for the monitored indicators, step 44. This analysis reveals the extent of coincidence of transitions of indicators during each time period, and is used to compute a transition correlation factor, step 46. As described in greater detail below, in some embodiments the transition correlation factor represents the ratio between the number of times the indicators transitioned together to the total number of indicators. However, other mathematical factors may be used

which capture the extent of correlation between transitions in indicator states, as will be recognized by those of skill in the art.

The transition factor is applied to the correlation factor to produce a new correlation factor, step 48. If the new correlation factor exceeds a threshold, step 50, indicating that the correlation is still sufficiently close to indicate the presence of the problem per the rule, the correlation engine generates and sends an alert message to the network management platform, step 52, so that the administrator becomes aware of the potential problem. This threshold may be predetermined from prior experience and set forth in the rule, or may be generated or adjusted from a predefined value based on values found in the transition correlation process described herein. For example, a predefined threshold value may be adjusted to account for a somewhat lower expected value of the correlation factor based on the number of indicators relevant to this transition analysis, the number of time slices in the window, the number of transitions, or other values. If the new correlation factor does not exceed the threshold, no such alert message is sent, step 54.

Figs. 4A-4B contain a flow chart showing one embodiment of this process used in conjunction with a specific correlation technique. In this technique, a window of time under consideration to determine whether a problem is occurring is divided into a number of discrete time slices, and a correlation factor for the indicators in the rule is generated for each time slice. The correlation factors are then combined over the entire window such as by calculating a mean correlation factor or a ratio of the number of times a correlation factor reached a threshold divided by the total number of time slices. This technique is employed, for example, in the Netcool®/Visionary™ program available from Micromuse Inc.

Referring to Fig. 4, the status monitors poll the monitored network devices or applications, step 60. The filters determine whether any of the monitored objects has reached a predefined severity level, step 62, and only allows the monitored event data to pass to the correlation engine when such level is reached to avoid overloading the engine with data. The correlation engine retrieves the one or more rules relating to the object for which a severity condition was reached, step 64, and further retrieves the state data relating to the other objects referenced in the rule, step 66.

Because the raw data collected by the status monitors is in disparate units specific to the device or application being monitored, e.g., number of instructions per second for a CPU, messages routed for a router, requests or hits for a server, etc., it is necessary to normalize the raw data so that an effective comparison may be done under the rule. Thus, severity mapping is performed on the collected state data, step 68, by converting the raw data to predefined states in accordance with a lookup table or similar mechanism for the device or application. The severity mapping may simply convert the raw data to one of two states, e.g., a high/low state or working/nonworking state, or may include additional gradations reflecting layers of severity.

A time window is defined during which the indicators in the rule are correlated, and this window is divided into sequential time slices, step 70. The length of the time window and time slices may be set as fixed amounts for all indicators or may be set in a particular rule, e.g., the rule defining how much indicators must correlate before correlation is found. As explained further herein, the edge correlation relies upon a concept of edges transitioning in the same timeslice. However, edges may not always transition in the same timeslice given varying polling intervals, response times from targets, etc. Thus, the engine adjusts the timing of the time slice being used to account for these measurement delays, step 70.

This adjustment may be performed in several different ways. One possible way is to increase the size of the timeslice to, e.g., six seconds, and to use a timeslice increment of three seconds when performing timeslice analysis. Another adjustment solution is for the correlation engine to examine, for each timeslice, all of the indicators that have changed in that timeslice. Of those that have changed, the engine starts with the one that has changed most recently and looks at the last changed time for all of the other indicators. If they have changed in the same direction in the past 5 seconds from the timestamp of the most recently changed indicator, they are considered to be correlated for the sake of calculating the edge factor. These solutions handle edges that fall in different timeslices.

To help better illustrate this point, the options are examined in turn. For all options, the following data set is used:

- Object 1: Transitions UP at time 2, DOWN at time 11, UP at time 18, DOWN at time 23
- Object 2: Transitions UP at time 4, DOWN at time 11, UP at time 16, DOWN at time 27

If no timing adjustment mechanism is used, with the above objects, and timeslices starting at t=1, two timeslices would be impacted, slice (20-24) when object 1 transitioned down and then slice 25-29 when object 2 transitioned down. If the timeslices started at timeslice 2, three timeslices would be impacted by a lack of edge correlation. The data for using no timing adjustment mechanism are summarized in Table I below:

TABLE I

Starting Time	Number of timeslices (out of 6) with edge factor < 1
T=0	2

T=1	2
T=2	4
T=3	3
T=4	3

If fractional slice increments are used, the first slice would be 0-5, and would see two edges go up. The next slice would be 3-8 and would see object 2 go up while object 1 stayed high. This would be an edge factor of 0.5, in accordance with the formulas provided below. The next slice would be 6-11 and would see both edges go down. The next slice, 9-14, would see both edges go down. The next slice, 12-17, would see object 2 go up and object 1 stay down. The next slice would be 15-20, and would see both edges go up. The next slice, 18-23 would see object 1 go down and object 2 stay high. The next slice, 21-26, has one edge go down. The next slice 24-29 has object 2 down, object 1 even.

This comes out to 9 slices, of which 5 showed no correlation. This number seems too high, to the extent it produces a result of 55% of the slices showing a lack of correlation, which does not appear to reflect the actual conditions. If the time slices start at t=1 instead of t=0, only 3 slices out of 9 are left not showing correlation. Starting at t=3 results in 5 slices without correlation, which represents too high and too wide a variance from other starting point configurations. Data on some results is summarized in Table II below.

TABLE II

Starting Time	Number of timeslices (out of 9) with edge factor < 1
T=0	5
T=1	3
T=2	4

Under the "last event time" method, if the time slices start at time = 0, and have timeslices at t=5, 10, 15, 20, 25, 30, then there is only one timeslice (t=20-24) when an edge transitioned and was not considered correlated. In this timeslice, Object 1 transitioned down, but Object 2 was still UP. At t=25-29, Object 2 transitioned down, but Object 1 appears to have transitioned down only 4 seconds earlier. Therefore it may be counted as transitioning down for the purposes of calculating the edge factor for this timeslice.

If the timeslices start at t=2, additional timeslices are at t=7, 12, 17, 22, 27, 32. In the first timeslice (2-6) both edges went up. In the second timeslice (7-11) both edges went down. In the third timeslice, (12-16), Object 2 went up and Object 1 showed a transition down 5 seconds earlier. This would create an edge factor of 0.5 for this timeslice. In the fourth timeslice (17-21) there were no edges. In the fourth timeslice (22-26) Object 1 transitioned down and object 2 remained high. An edge factor of 0.5 is calculated for this timeslice. In the next timeslice, (27-31) Object 2 transitioned down, and Object 1 had transitioned down four seconds earlier. Therefore they are both in correlation, and the edge factor is 1 for this timeslice.

Use of the last event time timing adjustment technique provide the results shown in Table III below.

TABLE III

Starting Time	Number of timeslices (out of 6) with edge factor < 1
t=0	1
t=1	1
t=2	2
t=3	1
t=4	1

The proper handling of timing adjustment maintains the benefits of edge correlation's noise reduction while avoiding the inaccuracies of arbitrary timeslice boundaries. Using the "last event time" mechanism appears to give the most benefit with the least variance due to timeslice boundary location.

Returning to Fig. 4A, the edge correlation routine begins the process of correlating objects based on their transitions. It selects those of the objects in the rule which have been assigned edge relevance, step 74. That is, a factor is assigned to each object in the rule which sets whether its transitions are relevant and should be included in an edge correlation analysis. For example, there are some rules written that use the presence of an indicator as part of the rule, but this indicator does not transition and as such would always detract from the rule's firing if edge correlation were applied to it. Edge relevance for each object is stored in the rule.

The following analysis is then performed for each time slice in the window, step 76. The correlation engine computes a correlation factor based on the state values of relevant or selected objects, step 78. This correlation factor may be an average of the state values, a weighted average, or any other desired computation set forth in the rule. For each selected object, step 80 (Fig. 4B), the object's state in the current time slice is analyzed to determine whether the object's state has remained the same, gone up, or gone down, step 84, as these time slices may have been adjusted as described above. This analysis may be performed using known techniques for detecting transitions in the state of an object, such as based upon frequent polling or measuring usage data, as further described in commonly owned patent application serial no. 09/724,025, attorney docket no. 3882/3, titled "METHOD AND SYSTEM FOR PREDICTING CAUSES OF NETWORK SERVICE OUTAGES USING TIME DOMAIN CORRELATION," which application is hereby incorporated by reference into this application. Alternatively, this

Express Mail No. EL718275755US

determination that a transition has occurred may be made by comparing the object's state during or at the end of the current time slice with its state at the end of the immediately preceding time slice, and detecting a transition if these two states are different. . If the severity scale allows for more than two states, this determination allows for a greater number of possible state changes. A count is maintained for each type of state change determination, step 86, and the next selected object is considered, step 88, until all relevant objects have been considered.

The edge correlation routine then computes an edge correlation factor, step 90, using the following formula:

Edge cf = Max(upcount, downcount, samecount) / (total number of relevant objects)

This formula provides that the edge correlation factor represents the percentage of the largest trend in object transitions over the total number of objects. The edge correlation factor is then multiplied by the correlation factor for the time slice, step 92, to produce a new correlation factor which is discounted from the old correlation factor to account for less than full correlation in the transition of objects during the time slice. This process is repeated for all time slices in the window, step 94, until all such slices have been processed.

As a result, a set of new correlation factors has been computed, one for each time slice. An overall correlation factor is then computed, step 96. This new overall correlation factor may be computed as the mean of the individual new correlation factors for the time slices. Alternatively, the overall factor may be found by determining in each time slice whether the slice's factor exceeds a predefined threshold and thus the rule applies or "fires," and then adding up the number of times the rule fired during the time window. In either case, the overall correlation factor is compared to an appropriate threshold, step 98, to determine whether an alert should be sent, step 100, or no alert should be sent, step 102.

In accordance with the invention, there may be logical groupings of indicators in which the components are expected to move together but the groups themselves are not expected to necessarily move together. This leads to embodiments of the present invention which account for edge correlation groups. The field in the rule used to indicate whether an indicator should be edge correlated may also be used to specify the edge correlation group to which the indicator belongs. For example, in the example show in Table IV, indicators 1-3 belong to group A, indicators 4-5 belong to group B, and indicators 6-7 have no edge relevance. The edge correlation factor would be computed separately for the "A" group and the "B" group, and no edge correlation is performed on indicators 6 and 7. Thus, if 2/3 of group A correlate and 1/2 of group B correlate, the edge factor would be 2/3*1/2=.333, instead of 3/5=0.600 if calculated without groupings.

TABLE IV

Indicator #	Edge Relevance/Group
1	A
2	A
3	A
4	B
5	B
6	N
7	N

Express Mail No. EL718275755US

BRMFS1 237890v5

Referring then to Fig. 5, a process for correlating edges with groups of indicators is shown. As described in greater detail above, the indicator values and severity levels are received, step 110, as are the appropriate rule(s), step 112. From the rules, the group or groups of indicators, if any, are determined, step 114. The use of grouping supports various logical operations among members of a group, allowing for differences in the treatment of indicators across groups during edge correlation. These logical operations are applied, step 116, to compute transition correlation factors for each group.

The correlation engine, in conjunction with the edge correlation routine, performs multivariable correlation for each time slice, including accounting for transitions by applying transitions correlation factors found on a group basis, step 118. If only one group is present in any given time slice, step 120, then the new correlation factor found for that group is the factor used for that time slice. If more than one group is present, then the new correlation factors found for each group are multiplied to produce the factor to be used for the time slice, step 122. Combining of these factors then proceeds as described earlier.

Figs. 6-11 illustrate various examples of the use of edge correlation to reduce the likelihood of false alarms. The formula used in these examples is essentially the same as set forth earlier:

$$\text{newcf} = (\max(\text{upindicators, downindicators, sameindicators})/\text{numindicators})_A * [(\max(\text{upindicators, downindicators, sameindicators})/\text{numindicators})_B] * \text{cf},$$

wherein:

newcf represents the new correlation factor, e.g., for a given time slice, after application of the edge correlation methologdy described herein;

upindicators, *downindicators* and *sameindicators* represent the number of state changes of type up, down, or same (no change), respectively, for indicators having edge relevance in a rule, e.g., in a give time slice;

numindicators represents the total number of indicators having edge relevance in a rule, e.g., in a give time slice; and

cf represents the base correlation factor found using multivariable correlation techniques such as those described herein.

For these examples, the rule monitors three indicators, with each indicator having an edge relevancy factor (rf) of 1. The indicators will be either at severity level (sev) 1 or sev 0, and the sev-scale mapping maps a sev 1 to a value of 1, and a sev 0 to a value of 0. The computation performed by the edge correlation routine starts with timeslice 0 to see the transitions into timeslice 1, and calculate the meancf (average cf)on timeslices 1-10.

Fig. 6 demonstrates that a rule firing at 100% with no edges is not impacted by the edge correlation computation.

Fig. 7 demonstrates that a rule with all indicators tracking is not impacted by edge correlation.

Fig. 8 demonstrates that the rule is weakened as a result of lack of complete transition correlation, in time slices 3, 7, and 10, but may still fire if the indicators track and there are not many edges.

Fig. 9 shows that in a situation where one indicator is pegged high, another alternates high and low, and the third alternates (mostly off), that the rule would have fired without edge correlation, but the use of edge correlation techniques as described herein reduced *cf* below the firing threshold.

Fig. 10 indicates that edges are needed in order for edge correlation to have the most impact. The rule would fire at 66.66% without edge correlation (as 2/3 of the indicators are always high) but only at 62.22% with edge correlation. The example demonstrates that edge correlation is most effective when indicators are transitioning more frequently.

Fig. 11 well illustrates the effectiveness of edge correlation. Simulating a noisy environment, it may be seen that the rule would otherwise have fired at a *meancf* of 80, but edge correlation reduces it down to a *newmeancf* of 51.11. If the window were larger and more edges present, this *newmeancf* would be even lower.

While the invention has been described and illustrated in connection with preferred embodiments, many variations and modifications as will be evident to those skilled in this art may be made without departing from the spirit and scope of the invention, and the invention is thus not to be limited to the precise details of methodology or construction set forth above as such variations and modification are intended to be included within the scope of the invention.